UFBA
Universidade
Federal da Bahia

Lógica Fuzzy

Implementação em Robocode

Disciplina: Inteligência Artificial

Lógica Fuzzy

Partindo da teoria...

Até que momento poderíamos dizer que o copo está vazio?

A partir de que momento específico este copo poderia ser considerado cheio?

Lógica Fuzzy

- Na área de computação, porém, é comum buscarmos representações precisas para modelar este tipo de cenário, desenvolvendo algoritmos que tentam quantificar estes fatores utilizando valores precisos.

- A lógica fuzzy, também conhecida como lógica nebulosa ou difusa, é uma técnica da área de inteligência computacional que nos permite representar modelos que contenham certo grau de incerteza ou imprecisão, características de situações do mundo real.

Lógica clássica vs. Lógica Fuzzy

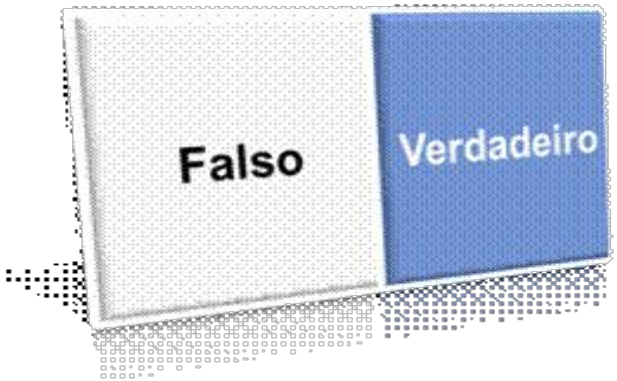
- Na lógica fuzzy, diferentemente da lógica clássica, um elemento pode pertencer **parcialmente** a um conjunto.

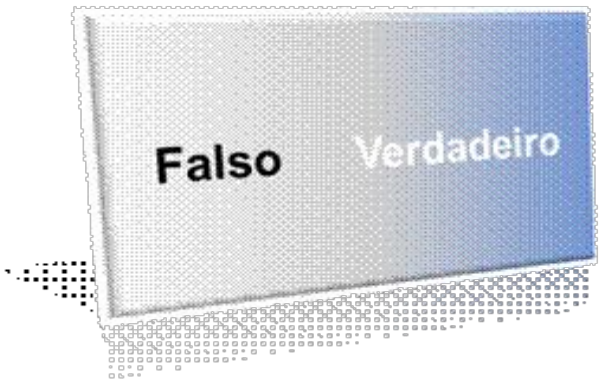
$$f_A(x) = \begin{cases} 1 & \text{se e somente se } x \in A \\ 0 & \text{se e somente se } x \notin A \end{cases}$$

$$\mu_A(x) : X \rightarrow [0,1]$$

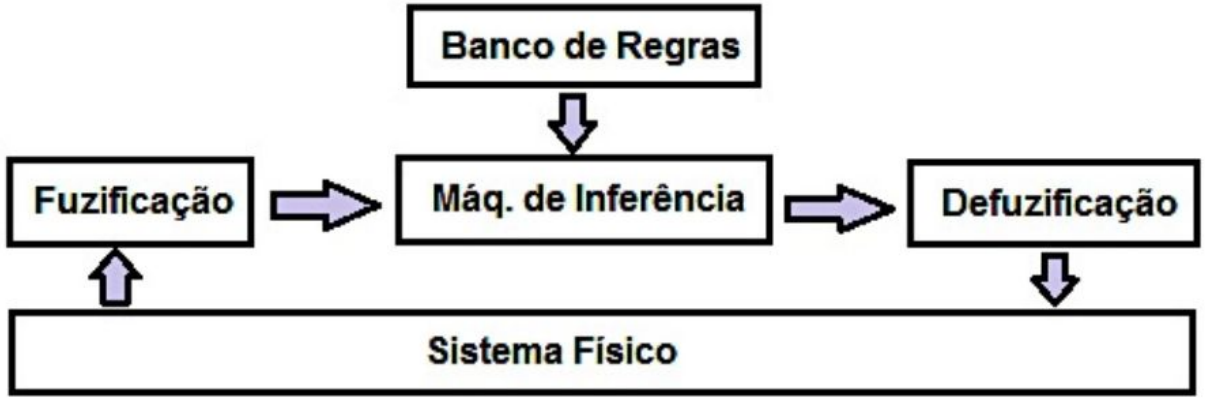
LÓGICA CLÁSSICA

LÓGICA NEBULOSA



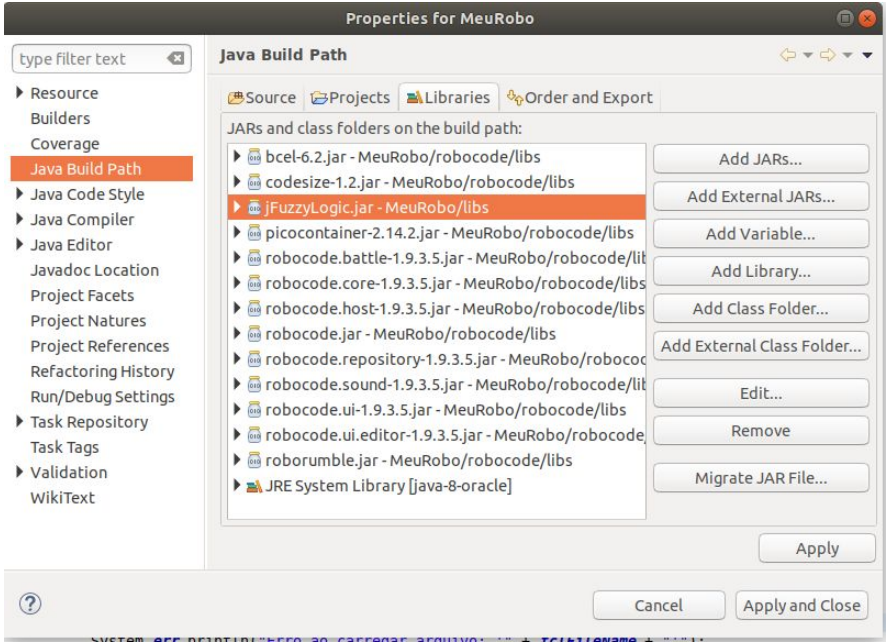


Sistemas Fuzzy



Aplicação no Robocode

- Vamos utilizar a biblioteca *jFuzzyLogic*, que será integrada ao código Java. Aqui, estou utilizando a IDE Eclipse.
  - Download: https://sourceforge.net/projects/jfuzzylogic/
- Adicione o arquivo **.jar** ao projeto.



Aplicação no Robocode

- Para criar nosso robô que utiliza a lógica fuzzy, é necessário criar um arquivos de regras.
- Esse arquivo possui a extensão ***.fcl***

Estrutura do arquivo de regras

Variáveis de entrada

Variáveis de saída

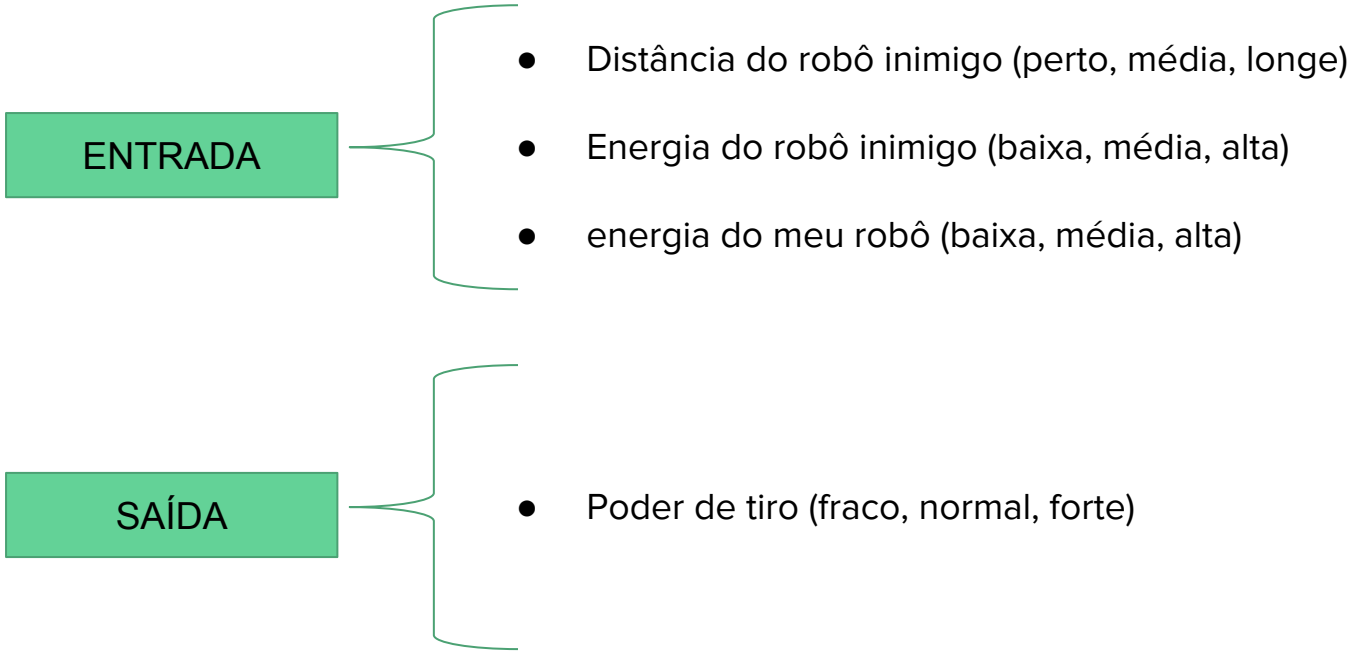
Fuzzification

Defuzzification

Conjunto de regras

```
FUNCTION_BLOCK simulador    // Início do bloco de definições

VAR_INPUT               // Definição das variáveis de entrada
    nome_variavel_entrada: REAL;
END_VAR

VAR_OUTPUT              // Definição das variáveis de saída
    nome_variavel_saida: REAL;
END_VAR

FUZZIFY nome_variavel_entrada
    // definição das partições fuzzy e seus intervalos para cada variável de entrada
    TERM PARTICAO_X := (0.0, 0) (10.0, 1) (20.0, 0)
END_FUZZIFY

DEFUZZIFY nome_variavel_saida
    // definição das partições fuzzy e seus intervalos para cada variável de saída

    TERM PARTICAO_Y := (0.0, 0) (50.0, 1) (100.0, 0) ;
    METHOD : COG;  // Método de defuzzificação (Padrão é o Centro de Gravidade)
    DEFAULT := 0;  // Valor default caso nenhuma regra seja ativada
END_DEFUZZIFY

RULEBLOCK No1
    // Definição do conjunto de regras para o controlador Fuzzy. Este bloco irá descrever
    // as correlações entre as partições da variável de entrada com uma partição da variável
    // de saída

    AND : MIN;  // Método MIN utilizado no processamento do operador lógico AND
    ACT : MIN;  // Método de ativação
    ACCU : MAX; // método de acumulação

    // Início da descrição de cada regra
    // RULE 1 : IF variavel_entrada1 IS PARTICAO1 AND variavel_entrada1 IS particao2  THEN variavel_saida IS particaoX;

END_RULEBLOCK

END_FUNCTION_BLOCK
```

Robocode: Aplicação para definir poder de tiro



Arquivo de regras: Definindo variáveis de entrada e saída

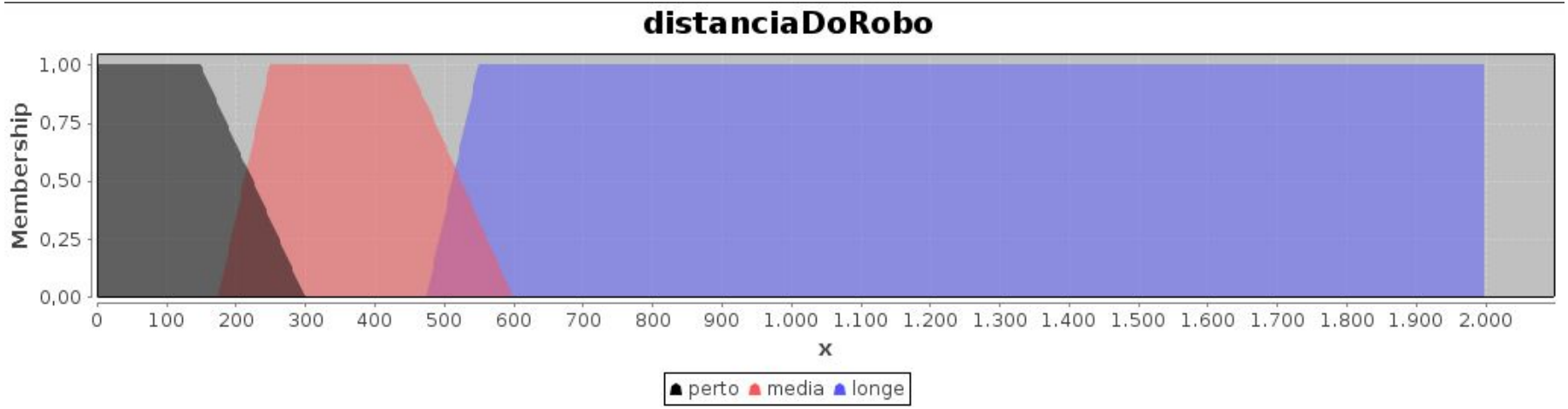
```
FUNCTION_BLOCK poder_tiro //Definição de bloco (pode haver mais de um bloco por arquivo)

VAR_INPUT               // Definição das variáveis de entrada
    distanciaDoRobo : REAL;
    energiaInimiga : REAL;
    minhaEnergia : REAL;
END_VAR

VAR_OUTPUT              // Definir a variável de saída
    poder : REAL;       // 0 à 3
END_VAR
```

...

Arquivo de regras: Definindo interface fuzzification

```
FUZZIFY distanciaDoRobo // Variáveis de entrada para Fuzzify 'distanciaDoRobo': {'perto', 'média' , 'longe'}
    TERM perto := trape 0 0 150 300;
    TERM media := trape 175 250 450 600;
    TERM longe := trape 475 550 2000 2000;
END_FUZZIFY
```

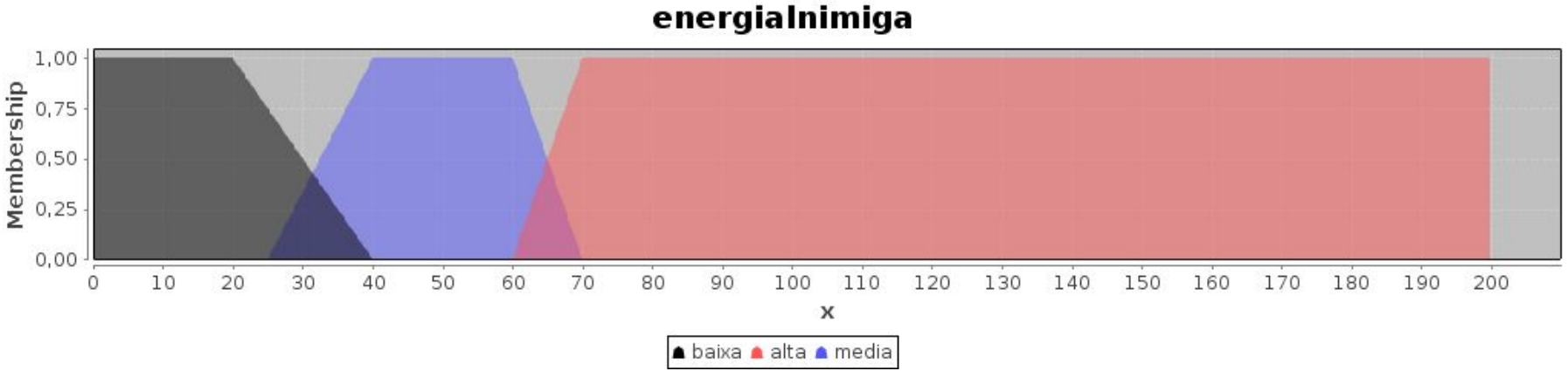
...



Arquivo de regras: Definindo interface fuzzification

```
FUZZIFY energiaInimiga  //  Variáveis de entrada para Fuzzify 'energiaInimiga': { 'baixa', 'media', 'alta' }
    TERM baixa := trape 0 0 20 40;
    TERM media := trape 25 40 60 70;
    TERM alta := trape 60 70 200 200;
END_FUZZIFY
```

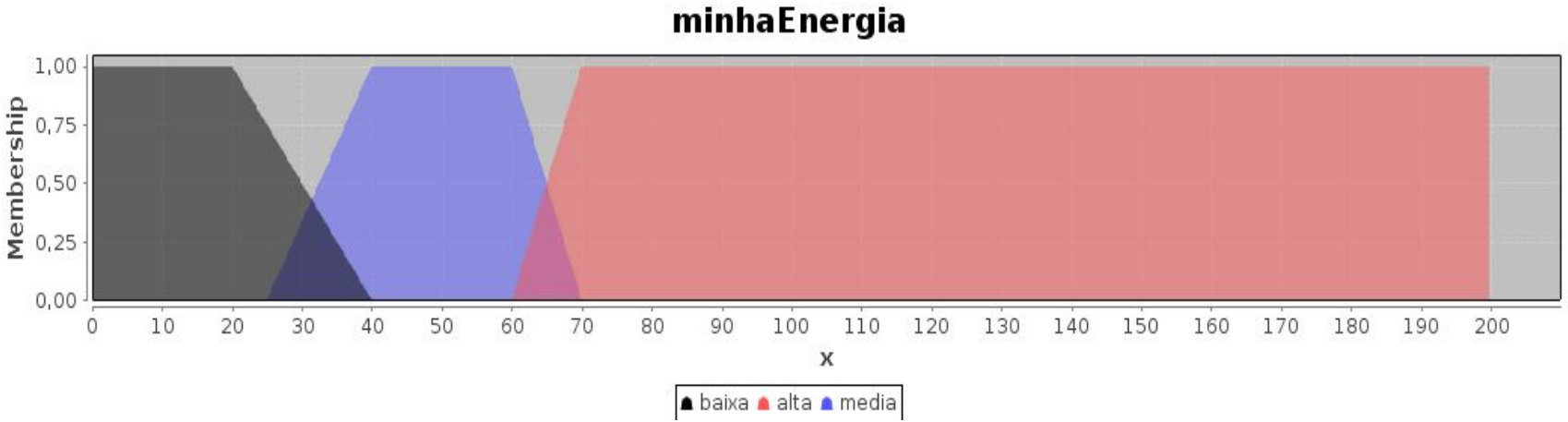
...



Arquivo de regras: Definindo interface fuzzification

```
FUZZIFY minhaEnergia    // Variáveis de entrada para Fuzzify  'minhaEnergia': { 'baixa', 'media', 'alta' }
    TERM baixa := trape 0 0 20 40;
    TERM media := trape 25 40 60 70;
    TERM alta := trape 60 70 200 200;
END_FUZZIFY
```

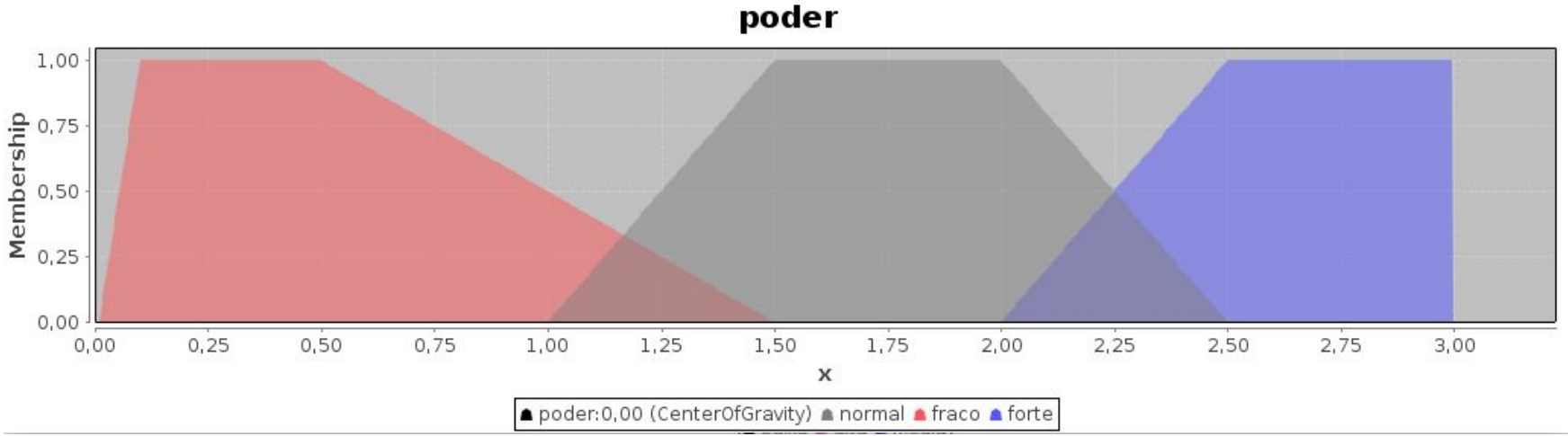
...



Arquivo de regras: Definindo interface defuzzification

```
DEFUZZIFY poder          // Variável de saída Defuzzify 'poder' : {'fraco', 'normal', 'forte' }
    TERM fraco := trape 0.01 0.1 0.5 1.5;
    TERM normal := trape 1 1.5 2 2.5;
    TERM forte := trape 2 2.5 3 3;
    METHOD : COG;        // Método 'Centro da gravidade'
    DEFAULT := 1;        // Valor default 1 (se nenhuma das regras ativar o defuzzifier)
END_DEFUZZIFY
```

...



Arquivo de regras: bloco de regras

IF ... **IS** .... **AND** ... **IS** ... **THEN** ... **IS** ....

```
RULEBLOCK No1
    AND : MIN;          // É comum utilizar o método MIN para o operador AND, e o método MAX para o operador OR.
    ACT : MIN;          // Use 'min' activation method
    ACCU : MAX;         // Use 'max' accumulation method

    RULE 1 : IF distanciaDoRobo IS perto AND NOT minhaEnergia IS baixa THEN poder IS forte;
    RULE 2 : IF distanciaDoRobo IS perto AND minhaEnergia IS baixa THEN poder IS normal;
    RULE 3 : IF distanciaDoRobo IS media AND minhaEnergia IS baixa THEN poder IS fraco;
    RULE 4 : IF distanciaDoRobo IS media AND minhaEnergia IS media THEN poder IS normal;
    RULE 5 : IF distanciaDoRobo IS media AND minhaEnergia IS alta AND NOT minhaEnergia IS alta THEN poder IS forte;
    RULE 6 : IF distanciaDoRobo IS longe AND minhaEnergia IS alta AND minhaEnergia IS baixa THEN poder IS normal;
    RULE 7 : IF distanciaDoRobo IS longe THEN poder IS fraco;
END_RULEBLOCK
```

...

```
FUNCTION_BLOCK poder_tiro //Definição de bloco (pode haver mais de um bloco por arquivo)

VAR_INPUT               // Definição das variáveis de entrada
    distanciaDoRobo : REAL;
    energiaInimiga : REAL;
    minhaEnergia : REAL;
END_VAR

VAR_OUTPUT              // Definir a variável de saída
    poder : REAL;       // 0 à 3
END_VAR

FUZZIFY distanciaDoRobo // Variáveis de entrada para Fuzzify 'distanciaDoRobo': {'perto', 'média' , 'longe'}
    TERM perto := trape 0 0 150 300;
    TERM media := trape 175 250 450 600;
    TERM longe := trape 475 550 2000 2000;
END_FUZZIFY

FUZZIFY energiaInimiga  //  Variáveis de entrada para Fuzzify 'energiaInimiga': { 'baixa', 'media', 'alta' }
    TERM baixa := trape 0 0 20 40;
    TERM media := trape 25 40 60 70;
    TERM alta := trape 60 70 200 200;
END_FUZZIFY

FUZZIFY minhaEnergia    // Variáveis de entrada para Fuzzify  'minhaEnergia': { 'baixa', 'media', 'alta' }
    TERM baixa := trape 0 0 20 40;
    TERM media := trape 25 40 60 70;
    TERM alta := trape 60 70 200 200;
END_FUZZIFY

DEFUZZIFY poder         // Variável de saída Defuzzify 'poder' : {'fraco', 'normal', 'forte' }
    TERM fraco := trape 0.01 0.1 0.5 1.5;
    TERM normal := trape 1 1.5 2 2.5;
    TERM forte := trape 2 2.5 3 3;
    METHOD : COG;       // Método 'Centro da gravidade'
    DEFAULT := 1;       // Valor default 1 (se nenhuma das regras ativar o defuzzifier)
END_DEFUZZIFY

RULEBLOCK No1
    AND : MIN;          // É comum utilizar o método MIN para o operador AND, e o método MAX para o operador OR.
    ACT : MIN;          // Use 'min' activation method
    ACCU : MAX;         // Use 'max' accumulation method

    RULE 1 : IF distanciaDoRobo IS perto AND NOT minhaEnergia IS baixa THEN poder IS forte;
    RULE 2 : IF distanciaDoRobo IS perto AND minhaEnergia IS baixa THEN poder IS normal;
    RULE 3 : IF distanciaDoRobo IS media AND minhaEnergia IS baixa THEN poder IS fraco;
    RULE 4 : IF distanciaDoRobo IS media AND minhaEnergia IS media THEN poder IS normal;
    RULE 5 : IF distanciaDoRobo IS media AND minhaEnergia IS alta AND NOT minhaEnergia IS alta THEN poder IS forte;
    RULE 6 : IF distanciaDoRobo IS longe AND minhaEnergia IS alta AND minhaEnergia IS baixa THEN poder IS normal;
    RULE 7 : IF distanciaDoRobo IS longe THEN poder IS fraco;
END_RULEBLOCK

END_FUNCTION_BLOCK
```

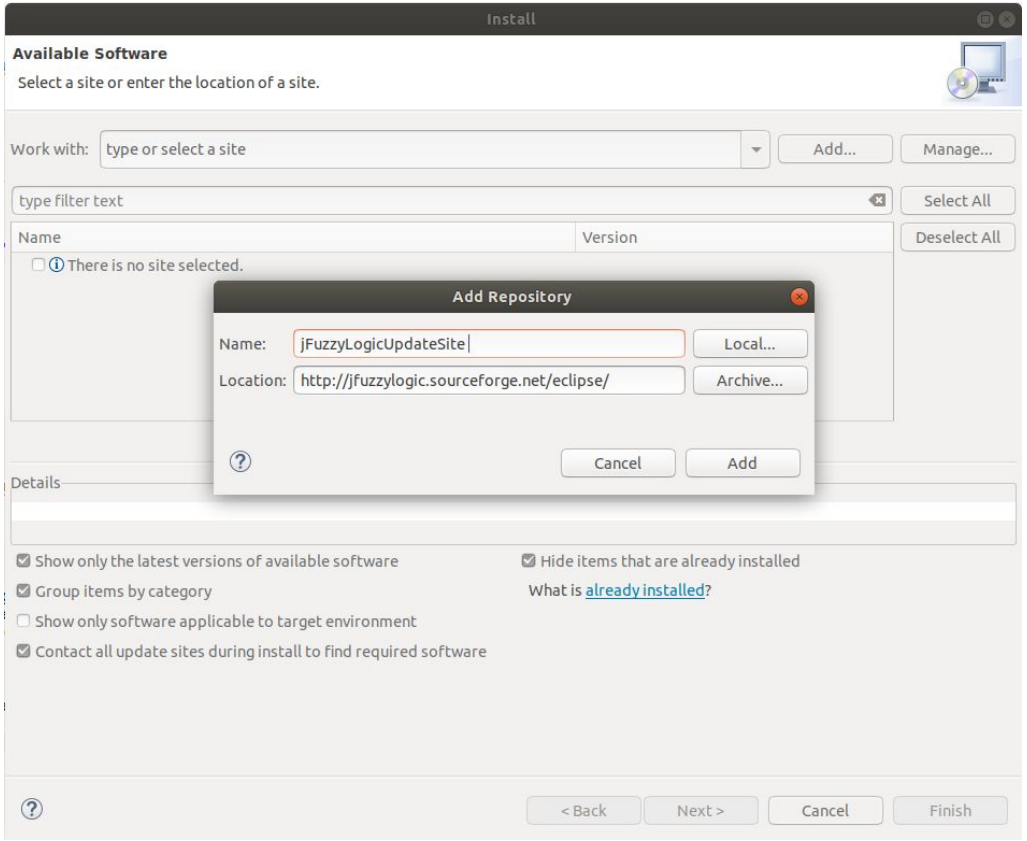
Arquivo de regras *.fcl*

Bloco de regras: Visualizar gráficos

Plugin Eclipse:

- Menu Help -> Install new software
- Clique em Add:
- Adicione:

Name: jFuzzyLogicUpdateSite
Location: http://jfuzzylogic.sourceforge.net/eclipse/



Integrando no Java

- Diretório do arquivo de regras

```
final static String fclFileName = "/home/joao/Documentos/workspace/MeuRobo/src/time/rules.fcl";
```

- Variável para chamar a função de regras

```
private FunctionBlock poderTiro;
```

Integrando no Java

Variável do jFuzzyLogic

Carregamento do arquivo de regras

Carregamento do bloco de funções

```
/Instancia o controlador a partir das definições do arquivo FLC
FIS fis = null;

@Override
public void run() {

    setColors(Color.green,Color.red,Color.red); // body,gun,radar
    fis = FIS.load(fclFileName);

    if( fis == null ) {
        System.err.println("Erro ao carregar arquivo: '" + fclFileName + "'");
        return;
    }

    poderTiro = fis.getFunctionBlock("poder_tiro");

    while (true) {
        this.setAhead(50);
        this.setTurnLeft(45);
        this.execute();
    }
}
```

Integrando no Java

Seta variáveis de entrada

Processa os cálculos para a inferência.

Obtenção do valor calculado.

```
//Executado quando o radar do seu robô encontra um adversário.
@Override
public void onScannedRobot(ScannedRobotEvent event) {
    System.out.println("Teste ");

    if(isAim(event)){
        poderTiro.setVariable("distanciaDoRobo", event.getDistance());
        poderTiro.setVariable("energiaInimiga", event.getEnergy());
        poderTiro.setVariable("minhaEnergia", getEnergy());

        poderTiro.evaluate();

        double poder = poderTiro.getVariable("poder").getValue();

        setFire(poder);

        System.out.println("Saída - "+poder);

    }

}
```

Material de apoio

Código fonte e apresentação:

- GitHub: https://github.com/joaobarros05/robocodeFuzzy

- Material para se aprofundar mais:
  - Introdução à Lógica Fuzzy com Java: https://www.devmedia.com.br/introducao-a-logica-fuzzy-com-java/32444